

MA8020 Tekniska beräkningar

Något om numerisk linjär algebra

Mikael Hindgren



27 november 2025

Linjära ekvationssystem

Vi vill lösa linjära ekvationssystem $Ax = b$:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases} \Leftrightarrow \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \\ a_{m1} & a_{12} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \\ b \end{pmatrix}$$

A

- Uppkommer t.ex. vid diskretisering av partiella differentialekvationer
- Antalet obekanta är ofta stort: $n = 10^5 \rightarrow 10^8$

Det är viktigt att kunna lösa linjära ekvationssystem effektivt och noggrann!

Terminologi:

- A kallas **koefficientmatris**, b **högerled** och x **lösningsvektor**
- Om $n > m/n < m$ är systemet **under-/överbestämt**
- Om $b = \mathbf{0}$ är ekvationssystemet **homogent**

Vi kommer att koncentrera oss på **kvadratiska ekvationssystem** ($m = n$).

Linjära ekvationssystem

Vilka kvadratiska ekvationssystem har entydig lösning?

Definition 1

Om matrisen A har linjärt oberoende kolonner kallas A **regulär** eller **icke-singulär**.

Sats 1

Matrisen A är icke-singulär om

- $Ax = \mathbf{b}$ har entydig lösning för varje högerled \mathbf{b}
- $\det A \neq 0$
- Inversen A^{-1} existerar

Sats 2

Om matrisen A är kvadratisk har ekvationssystemet $Ax = \mathbf{b}$ entydig lösning, ingen lösning eller oändligt många lösningar.

Anm: Om A^{-1} existerar har $Ax = \mathbf{b}$ den entydiga lösningen $x = A^{-1}\mathbf{b}$.

Linjära ekvationssystem

IIIa-konditionering

- Om $\det A \approx 0$ är ekvationssystemet **illa-konditionerat**. Små förändringar av matriselementen i A eller i b kan då ge stora förändringar i lösningen.
- Om $\det A = 0$ är A **singulär**.

Exempel 1

En välkonditionerad (A_1) och en illa-konditionerad (A_2) matris:

```
Remove["Global`*"]
A1 = {{0.0001, 1}, {1, 1}};
A2 = {{1, 1}, {1, 1.0001}};
x = {x1, x2};
b1 = {2, 2};
b2 = {2, 2.0001};
Print["A1 = ", MatrixForm[A1], ", Det A1 = ", Det[A1], ", A1x = b1: ",
  Solve[A1.x == b1, x][[1]], ", A1x = b2: ", Solve[A1.x == b2, x][[1]]];
Print["A2 = ", MatrixForm[A2], ", Det A2 = ", Det[A2], ", A2x = b1: ",
  Solve[A2.x == b1, x][[1]], ", A2x = b2: ", Solve[A2.x == b2, x][[1]]];
A1 = {{0.0001, 1}, {1, 1}}, Det A1 = -0.9999, A1x = b1: {x1 → 0., x2 → 2.}, A1x = b2: {x1 → 0.00010001, x2 → 2.}
A2 = {{1, 1}, {1, 1.0001}}, Det A2 = 0.0001, A2x = b1: {x1 → 2., x2 → -6.44689×10-17}, A2x = b2: {x1 → 1., x2 → 1.}
```

Linjära ekvationssystem

Lösningsmetoder för linjära ekvationssystem

- Två typer av metoder används för att lösa linjära ekvationssystem numeriskt:
Direkta och **iterativa**.
- Den vanligaste direkta metoden är **Gausselimination** som utnyttjar att lösningen inte ändras under **elementära radoperationer** dvs om:
 - 1 Två rader byter plats
 - 2 En rad multipliceras med en konstant
 - 3 En rad adderas till en annan
- Om ett ekvationssystem kan omformas till ett annat genom (1) - (3) har de samma lösning och kallas **radekvivalenta**.

Linjära ekvationssystem

Gausselimination

Exempel 2

Vi vill lösa evationssystemet med Gausselimination:

$$\begin{cases} x_1 - 2x_2 + 2x_3 = 1 & (\text{R1}) \\ 3x_1 + x_2 - 2x_3 = -2 & (\text{R2}) \\ 2x_1 + x_2 - 2x_3 = -3 & (\text{R3}) \end{cases}$$

Gausselimination

- ➊ Triangulering: Nollställ elementen under diagonalelementen i varje kolonn med elementära radoperationer
 - ➋ Gör bakåtsubstitution
- ➌ Metoden innebär division med diagonalelement, som därför måste vara $\neq 0$ (helst inte heller nära noll).
 - ➍ Radomkastning placerar det tal i diagonalen som har störst absolutbelopp (**pivåelementet**) i den kolonn som skall nollställas.
 - ➎ Om inget pivåelement $\neq 0$ finns avbryts elimineringen och lösning saknas.

Linjära ekvationssystem

Gausselimination

Exempel 2 (forts)

Nollställning i K1 av R2 & R3. R2 är pivotrad och R1 & R2 kastas om. En kopia av R1 multipliceras med $\frac{1}{3}$ för R2 och $\frac{2}{3}$ för R3 och subtraheras från R2 & R3:

$$\left(\begin{array}{ccc|c} 1 & -2 & 2 & 1 \\ 3 & 1 & -2 & -2 \\ 2 & 1 & -2 & -3 \end{array} \right) \Leftrightarrow \left(\begin{array}{ccc|c} 3 & 1 & -2 & -2 \\ 1 & -2 & 2 & 1 \\ 2 & 1 & -2 & -3 \end{array} \right) \Leftrightarrow \left(\begin{array}{ccc|c} 3 & 1 & -2 & -2 \\ 0 & -\frac{7}{3} & \frac{8}{3} & \frac{5}{3} \\ 0 & \frac{1}{3} & -\frac{2}{3} & -\frac{5}{3} \end{array} \right)$$

Nollställning i K2 av R3. R2 är pivotrad ($|- \frac{7}{3}| > |\frac{1}{3}|$) och omkastning behövs ej.

En kopia av R2 multipliceras med $\frac{\frac{1}{3}}{-\frac{7}{3}} = -\frac{1}{7}$ och subtraheras från R3:

$$\Leftrightarrow \left(\begin{array}{ccc|c} 3 & 1 & -2 & -2 \\ 0 & -\frac{7}{3} & \frac{8}{3} & \frac{5}{3} \\ 0 & \frac{1}{3} & -\frac{2}{3} & -\frac{5}{3} \end{array} \right) \Leftrightarrow \left(\begin{array}{ccc|c} 3 & 1 & -2 & -2 \\ 0 & -\frac{7}{3} & \frac{8}{3} & -\frac{5}{3} \\ 0 & 0 & -\frac{2}{7} & -\frac{10}{7} \end{array} \right)$$

Bakåtsubstitution:

$$x_3 = \frac{-\frac{10}{7}}{-\frac{2}{7}} = 5, \quad x_2 = \frac{\frac{5}{3} - \frac{8}{3} \cdot 5}{-\frac{7}{3}} = 5, \quad x_1 = \frac{-2 + 2 \cdot 5 - 1 \cdot 5}{3} = 1.$$

Linjära ekvationssystem

Gausselimination

Anm:

- Gausselimination är den direkta metod som kräver minst antal flyttalsoperationer för att lösa ett givet ekvationssystem.
- Lösningstiden ges av $T = kn^3$ där n är antalet obekanta och k en datorberoende konstant.
- Om man behöver lösa flera ekvationssystem av typen

$$A\mathbf{x} = \mathbf{b}_i, \quad i = 1, 2, \dots, k,$$

är det effektivare att använda två triangulära matriser.

- Matrisen A skrivs som $A = LU$ där U är övertriangulär och L undertriangular. Denna uppdelning av A kallas **LU-faktorisering**.
- Metod:
 - 1 Bestäm U och L så att $A = LU$.
 - 2 Inför en ny variabel $\mathbf{y} = U\mathbf{x}$ och lös för varje \mathbf{b}_i först $L\mathbf{y} = \mathbf{b}_i$ och sedan $U\mathbf{x} = \mathbf{y}$.

Eftersom L och U är triangulära är båda ekvationssystemen i (2) enkla att lösa.

Linjära ekvationssystem

Iterativa lösningsmetoder

Iterativa metoder är speciellt lämpliga för stora **glesa system** där de flesta matriselementen i systemmatrisen A är noll.

Metod:

- Skriv A som $A = B + (A - B)$ där B är en reguljär matris som är lätt att invertera och sådan att $A - B$ kan betraktas som en "liten störning" av B :

$$Ax = b \Leftrightarrow Bx = (B - A)x + b \Leftrightarrow x = B^{-1}((B - A)x + b) = (I - B^{-1}A)x + B^{-1}b$$

- Sambandet kan nu skrivas som en fixpunktsiteration (jfr $x_{k+1} = g(x_k)$):

$$\mathbf{x}^{(k+1)} = (I - B^{-1}A)\mathbf{x}^{(k)} + B^{-1}\mathbf{b}$$

Linjära ekvationssystem

Iterativa lösningsmetoder

Frågetecken:

- Hur väljer vi startlösningen $\mathbf{x}^{(0)}$?
- Vad krävs för att fixpunktsiterationen ska konvergera?

Sats 3

- Om A och B är reguljära matriser så konvergerar fixpunktsiterationen

$$\mathbf{x}^{(k+1)} = (I - B^{-1}A)\mathbf{x}^{(k)} + B^{-1}\mathbf{b}$$

mot lösningen till $A\mathbf{x} = \mathbf{y}$ omm egenvärdena λ_i till matrisen $I - B^{-1}A$ uppfyller $\max(|\lambda_i|) < 1$.

- Felet i lösningen

$$|\Delta\mathbf{x}^{(k)}| = |\mathbf{x} - \mathbf{x}^{(k)}| \leq C \max(|\lambda_i|)^k$$

Vi kan alltså räkna med snabb konvergens om villkoren i satsen är uppfyllda.

Linjära ekvationssystem

Iterativa lösningsmetoder: Jacobis metod

Jacobis metod bygger på att vi delar upp A enligt $A = L + D + U$ där L är strikt undertriangulär, D diagonal och U strikt övertriangulär.

Med $B = D$ och $A - B = L + U$ kan fixpunktiterationen skrivas som

$$\mathbf{x}^{(k+1)} = D^{-1}(\mathbf{b} - (L + U)\mathbf{x}^{(k)}) \Leftrightarrow x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j \neq i}^n a_{ij} x_j^{(k)} \right)$$

Sats 4

Jacobis iterationsmetod konvergerar om A är diagonaldominant dvs om

$$|a_{ii}| > \sum_{j=1, j \neq i}^n |a_{ij}| \quad \text{för alla } i.$$

Anm:

- Ju mer diagonaldominant A är desto snabbare konvergens.
- D diagonal $\Rightarrow D = (a_{ij})$, $a_{ij} = 0$ för $i \neq j \Rightarrow D^{-1}$ diagonal med diagonalelement $(\frac{1}{a_{ii}})$.

Linjära ekvationssystem

Iterativa lösningsmetoder: Jacobis metod

Exempel 3

Lös ekvationssystemet

$$\begin{cases} 3x_1 - x_2 + x_3 = 1 \\ 2x_1 - 4x_2 + x_3 = 1 \quad \leftarrow A \text{ diagonaldominant!} \\ -x_1 + x_2 - 4x_3 = -1 \end{cases}$$

```

In[=]:= Remove["Global`*"]
a = {{3, -1, 1}, {2, -4, 1}, {-1, 1, -4}};
u = UpperTriangularize[a, 1];
d = DiagonalMatrix[Diagonal[a]];
l = a - d - u;
xold = {0, 0, 0};
b = {1, 1, -1};
kmax = 20;
For[k = 1, k <= kmax, ++k,
  xnew = Inverse[d].(b - (l + u).xold);
  xold = xnew;
]
Print["Jacobi: ", N[xnew]];
Print["Kontroll med NSolve: ", NSolve[a.{x1, x2, x3} == b, {x1, x2, x3}]]
Jacobi: {0.249976, -0.0833073, 0.166647}
Kontroll med NSolve: {{x1 → 0.25, x2 → -0.0833333, x3 → 0.166667}}

```

Linjära ekvationssystem

Iterativa lösningsmetoder: Gauss-Seidels metod

Denna metod bygger också på att $A = L + D + U$ men med en alternativ omskrivning:

$$(L + D + U)\mathbf{x} = \mathbf{b} \Leftrightarrow D\mathbf{x} = \mathbf{b} - L\mathbf{x} - U\mathbf{x}$$

Denna formel ger fixpunktsiterationen:

$$\mathbf{x}^{(k+1)} = D^{-1}(\mathbf{b} - L\mathbf{x}^{(k+1)} - U\mathbf{x}^{(k)}) \Leftrightarrow \mathbf{x}^{(k+1)} = (L + D)^{-1}(\mathbf{b} - U\mathbf{x}^{(k)})$$

Anm:

- I Gauss-Seidels metod använder vi $B = L + D$ medan Jacobis metod använder $B = D$.
- Fixpunktsiterationen använder även det "nyaste" $\mathbf{x}^{(k+1)}$ i högerledet och vi kan därför förvänta oss snabbare konvergens än med Jacobis metod.
- Ett tillräckligt villkor för konvergens är även här att A är diagonaldominant.

Linjära ekvationssystem

Iterativa lösningsmetoder: Gauss-Seidels metod

Exempel 4

Samma ekvationssystem igen:

$$\begin{cases} 3x_1 - x_2 + x_3 = 1 \\ 2x_1 - 4x_2 + x_3 = 1 \\ -x_1 + x_2 - 4x_3 = -1 \end{cases}$$

```

In[+]:= Remove["Global`*"]
a = {{3, -1, 1}, {2, -4, 1}, {-1, 1, -4}};
u = UpperTriangularize[a, 1];
d = DiagonalMatrix[Diagonal[a]];
l = a - d - u;
b = {1, 1, -1};
xold = {0, 0, 0};
kmax = 20;
For[k = 1, k ≤ kmax, ++k,
  xnew = Inverse[d].(b - (l + u).xold);
  xold = xnew;
]
Print["Jacobi: ", N[xnew]];
xold = {0, 0, 0};
For[k = 1, k ≤ kmax, ++k,
  xnew = Inverse[l + d].(b - u.xold);
  xold = xnew;
]
Print["Gauss-Seidel: ", N[xnew]];
Print["NSolve: ", NSolve[a.{x1, x2, x3} == b, {x1, x2, x3}]]
Jacobi: {0.249976, -0.0833073, 0.166647}
Gauss-Seidel: {0.25, -0.0833333, 0.166667}
NSolve: {{x1 → 0.25, x2 → -0.0833333, x3 → 0.166667}}

```

Linjära ekvationssystem

Iterativa lösningsmetoder: Jacobis vs Gauss-Seidels metoder

Egenskap	Jacobi	Gauss-Seidel
Matrisformel	$\mathbf{x}^{(k+1)} = D^{-1}(\mathbf{b} - (L + U)\mathbf{x}^{(k)})$	$\mathbf{x}^{(k+1)} = (L + D)^{-1}(\mathbf{b} - U\mathbf{x}^{(k)})$
Iterationsformel	Använder $\mathbf{x}^{(k)}$ för <i>alla</i> variabler i iteration $k + 1$.	Använder de <i>senast beräknade</i> värdena $\mathbf{x}^{(k+1)}$ i samma iteration.
Uppdatering	<i>Globalt</i> : Hela vektorn uppdateras samtidigt.	<i>Sekventiellt</i> : Komponenter uppdateras efter hand (in-place).
Konvergens	Vanligtvis <i>långsammare</i> .	Vanligtvis <i>snabbare</i> (upp till dubbelt så snabb).
Parallelisering	Mycket <i>lätt att parallelisera</i> . Beräkningen av varje x_i är oberoende.	<i>Svårare att parallelisera</i> på grund av sekventiella beroenden.
Bäst när?	När <i>parallelitet</i> (t.ex. GPU-beräkning) är avgörande.	När seriell beräkning används och snabb konvergens är viktigast.

Sammanfattning:

- Jacobi är bäst när man har tillgång till en parallell arkitektur (t.ex. multi-core CPU eller GPU), eftersom beräkningarna för varje variabel kan göras helt samtidigt.
- Gauss-Seidel är bäst i seriella miljöer (enkel processor) där snabbast möjliga konvergens per iteration är önskvärd.

Egenvektorer och egenvärden

Definition 2

Vektorn $\mathbf{x} \neq \mathbf{0}$ är en egenvektor till matrisen A med egenvärdet λ om $A\mathbf{x} = \lambda\mathbf{x}$.

- Ekvationsystemet kan skrivas som $(A - \lambda I)\mathbf{x} = \mathbf{0}$.
- Det har icke-trivial lösning $\mathbf{x} \neq \mathbf{0}$ omm

$\det(A - \lambda I) = 0 \quad \leftarrow A\text{:s karakteristiska ekvation.}$

Exempel 5

Bestäm samtliga egenvärden och egenvektorer till $A = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$.

Karakteristisk ekvation:

$$\det(A - \lambda I) = \begin{vmatrix} 1 - \lambda & 2 \\ 2 & 4 - \lambda \end{vmatrix} = (1 - \lambda)(4 - \lambda) - 2 \cdot 2 = \lambda(\lambda - 5) = 0 \Leftrightarrow \begin{cases} \lambda_1 = 0 \\ \lambda_2 = 5. \end{cases}$$

Eigenvektorer och egenvärden

Exempel 5

Eigenvektorer:

$$\lambda_1 = 0 : (A - \lambda_1 I)\mathbf{x} = \mathbf{0}$$

$$\begin{aligned} &\Leftrightarrow \begin{cases} (1 - 0)x_1 + 2x_2 = 0 \\ 2x_1 + (4 - 0)x_2 = 0 \end{cases} \Leftrightarrow \begin{cases} x_1 + 2x_2 = 0 \\ 2x_1 + 4x_2 = 0 \end{cases} \\ &\Leftrightarrow (x_1, x_2) = t(-2, 1) \end{aligned}$$

$$\lambda_2 = 5 : (A - \lambda_2 I)\mathbf{x} = \mathbf{0}$$

$$\begin{aligned} &\Leftrightarrow \begin{cases} (1 - 5)x_1 + 2x_2 = 0 \\ 2x_1 + (4 - 5)x_2 = 0 \end{cases} \Leftrightarrow \begin{cases} -4x_1 + 2x_2 = 0 \\ 2x_1 - x_2 = 0 \end{cases} \\ &\Leftrightarrow (x_1, x_2) = s(1, 2) \end{aligned}$$

\therefore Alla vektorer som är parallella med $(-2, 1)$
 resp $(1, 2)$ är eigenvektorer till A med egenvärdet
 0 resp 5.

```
Remove["Global`*"]
A = {{1, 2}, {2, 4}};
Eigensystem[A] // MatrixForm
\left(\begin{array}{cc} 5 & 0 \\ \{1, 2\} & \{-2, 1\} \end{array}\right)
```

Eigenvektorer och egenvärden

Potensmetoden

Potensmetoden är en iterativ metod för att beräkna en eigenvektor. Antag att egenvärdena till A uppfyller

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$$

Om $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$ är en bas av eigenvektorer till A som svarar mot egenvärdena λ_i , $i = 1, 2, \dots, n$, kan startvektorn skrivas som

$$\mathbf{x}_0 = c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + \cdots + c_n \mathbf{v}_n$$

Eftersom $A\mathbf{v}_i = \lambda \mathbf{v}_i \Rightarrow A^k \mathbf{v}_i = \lambda^k \mathbf{v}_i$ får vi

$$A^k \mathbf{x}_0 = c_1 \lambda_1^k \mathbf{v}_1 + c_2 \lambda_2^k \mathbf{v}_2 + \cdots + c_n \lambda_n^k \mathbf{v}_n \Leftrightarrow \frac{A^k \mathbf{x}_0}{\lambda_1^k} = c_1 \mathbf{v}_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k \mathbf{v}_2 + \cdots + \left(\frac{\lambda_n}{\lambda_1}\right)^k \mathbf{v}_n$$

Om k är stort är $A^k \mathbf{x}_0$ nästan parallell med \mathbf{v}_1 eftersom $\left(\frac{\lambda_i}{\lambda_1}\right)^k \approx 0$ för $i \neq 1$.

$\therefore \mathbf{x}_k = A^k \mathbf{x}_0$ närmar sig en eigenvektor som motsvarar det egenvärde som har störst absolutbelopp.

Eigenvektorer och egenvärden

Potensmetoden

För att hindra att vektorerna blir för stora normalerar vi under varje iteration.

Potensmetoden

Välj en godtycklig startgissning \mathbf{x}_0 för eigenvektorn och iterera

- $\mathbf{y}_{k+1} = A\mathbf{x}_k$
- $\mathbf{x}_{k+1} = \mathbf{y}_{k+1}/|\mathbf{y}_{k+1}|$

tills $|\mathbf{x}_{k+1} - \mathbf{x}_k| < \delta$

Om vi har en eigenvektor kan vi beräkna motsvarande egenvärde:

$$A\mathbf{v} = \lambda\mathbf{v} \Leftrightarrow \mathbf{v}^T A\mathbf{v} = \lambda\mathbf{v}^T \mathbf{v} = \lambda|\mathbf{v}|^2 \Leftrightarrow \lambda = \frac{\mathbf{v}^T A\mathbf{v}}{|\mathbf{v}|^2} \quad \leftarrow \text{Rayleighkvot}$$

Vi får därför direkt en approximation till λ_1 som vi kan använda i iterationen ovan:

$$\lambda_1 = \frac{\mathbf{v}_1^T A\mathbf{v}_1}{|\mathbf{v}_1|^2} \approx \frac{\mathbf{y}_k^T A\mathbf{y}_k}{|\mathbf{y}_k|^2} = \frac{\mathbf{y}_k^T A\mathbf{x}_k}{|\mathbf{y}_k|} = \mathbf{x}_k^T \mathbf{y}_{k+1}$$

Eigenvektorer och egenvärden

Potensmetoden

Exempel 6

Bestäm största egenvärde och tillhörande egenvektor till $A = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & -3 \\ 1 & 0 & 1 \end{pmatrix}$

```

Remove["Global`*"]
A = {{3, 2, 1}, {2, 1, -3}, {1, 0, 1}};
xold = {1., 1., 1.};
For[k = 1, k ≤ 15, k++,
  ynew = A.xold;
  xnew = ynew / Norm[ynew];
  lambda = xold.ynew;
  xold = xnew;
]
Print["Normerad egenvektor: ", N[xnew]]
Print["Största egenvärde: ", N[lambda]]
Eigensystem[N[A]] // MatrixForm
Normerad egenvektor: {0.904535, 0.301502, 0.301517}
Största egenvärde: 3.99997
( 4. 2. -1.
 {0.904534, 0.301511, 0.301511} {0.57735, -0.57735, 0.57735} {-0.481543, 0.842701, 0.240772} )

```

Eigenvektorer och egenvärden

Invers iteration och skiftade potensmetoden

Hur gör vi om vi vill bestämma samtliga egenvärden och eigenvektorer?

Om A är inverterbar har vi

$$A\mathbf{x} = \lambda\mathbf{x} \Leftrightarrow A^{-1}A\mathbf{x} = \lambda A^{-1}\mathbf{x} \Leftrightarrow A^{-1}\mathbf{x} = \frac{1}{\lambda}\mathbf{x}$$

Slutsatser:

- A och A^{-1} har samma eigenvektorer.
- Om λ är egenvärde till A så är $\frac{1}{\lambda}$ egenvärde till A^{-1} vilket innebär att:
 A är inverterbar $\Leftrightarrow \lambda_i \neq 0, i = 1, 2, \dots, n$.
- Vill vi beräkna eigenvektorn som svarar mot det egenvärde som har *minst* absolutbelopp kan vi använda fixpunktsiterationen $\mathbf{x}_{n+1} = A^{-1}\mathbf{x}_n$.

Eigenvektorer och egenvärden

Invers iteration och skiftade potensmetoden

Exempel 7

Bestäm minsta egenvärde och tillhörande egenvektor till $A = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & -3 \\ 1 & 0 & 1 \end{pmatrix}$

```

Remove["Global`*"]
A = {{3, 2, 1}, {2, 1, -3}, {1, 0, 1}};
invA = Inverse[A];
xold = {1., 1., 1.};
For[k = 1, k ≤ 15, k++,
  ynew = invA.xold;
  xnew = ynew / Norm[ynew];
  lambda = xold.ynew;
  xold = xnew;
]
Print["Normerad egenvektor: ", N[xnew]]
Print["Minsta egenvärde: ", 1/N[lambda]]
Eigensystem[N[A]] // MatrixForm

Normerad egenvektor: {0.481548, -0.842702, -0.24076}
Minsta egenvärde: -0.99997
( 4. 2. -1.
 {0.904534, 0.301511, 0.301511} {0.57735, -0.57735, 0.57735} {-0.481543, 0.842701, 0.240772} )

```

Eigenvektorer och egenvärden

Invers iteration och skiftade potensmetoden

Hur beräknar vi egenvärdena mellan de som har störst och minst absolutbelopp?

Om s är ett tal har vi

$$(A - sl)\mathbf{x} = A\mathbf{x} - s\mathbf{x} = \lambda\mathbf{x} - s\mathbf{x} = (\lambda - s)\mathbf{x}$$

Slutsats:

- A och $A - sl$ har samma eigenvektorer.
- Om λ är ett egenvärde till A så är $\lambda - s$ ett egenvärde till $A - sl$.
- Med invers iteration kan vi beräkna det egenvärde $\lambda - s$ till $A - sl$ som har minst absolutbelopp dvs det egenvärde λ till A som ligger närmast s .

Eigenvektorer och egenvärden

Invers iteration och skiftade potensmetoden

Exempel 8

Bestäm egenvärdet till A som ligger närmast 1.5 samt tillhörande egenvektor.

$$A = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 1 & -3 \\ 1 & 0 & 1 \end{pmatrix}$$

```

Remove["Global`*"]
A = {{3, 2, 1}, {2, 1, -3}, {1, 0, 1}};
i = IdentityMatrix[3];
s = 1.5;
invAsI = Inverse[A - s * i];
xold = {1., 1., 1.};
For[k = 1, k <= 15, k++,
  ynew = invAsI.xold;
  xnew = ynew / Norm[ynew];
  lambda = xold.ynew;
  xold = xnew;
]
Print["Normerad egenvektor: ", N[xnew]]
Print["Egenvärdet närmast ", s, ": ", 1/N[lambda] + s]
Eigenvalues[N[A]] // MatrixForm
Normerad egenvektor: {0.57735, -0.57735, 0.57735}
Egenvärdet närmast 1.5: 2.

```

$$\left(\begin{array}{ccc} 4. & 2. & -1. \\ \{0.904534, 0.301511, 0.301511\} & \{0.57735, -0.57735, 0.57735\} & \{-0.481543, 0.842701, 0.240772\} \end{array} \right)$$

Eigenvektorer och egenvärden

Invers iteration och skiftade potensmetoden

Hur hittar vi startvärdena s ?

Gerschgorins sats

Om $A = (a_{ij})$ är en $n \times n$ -matris så är $C_i = \{z \in \mathbb{C} : |z - a_{ii}| \leq \sum_{j \neq i} |a_{ij}|\}$ cirklar i det komplexa talplanet. För varje egenvärde λ till A gäller då

$$\lambda \in \bigcup_{i=1}^n C_i$$

- Varje egenvärde ligger i minst en av de cirklarna.
- Varje diagonalelement a_{ii} är centrum för en cirkel och radens övriga element bestämmer cirkelns radie.
- Ju mer diagonaldominant matrisen är desto närmare diagonalelementen ligger egenvärdena.
- Om en grupp av k cirklar inte överlappar några andra cirklar innehåller den gruppen k egenvärden.
⇒ Ett specifikt egenvärde kan bara kopplas till en specifik cirkel om cirkeln är disjunkt från övriga.

Eigenvektorer och egenvärden

Invers iteration och skiftade potensmetoden

Exempel 9

Matrisen

$$A = \begin{pmatrix} 1 & 6 & -1 \\ 3 & -2 & 1 \\ -3 & 6 & -5 \end{pmatrix}$$

har egenvärdena $\lambda_1 = -8$, $\lambda_2 = 4$ och $\lambda_3 = -2$.

Vi får:

$$C_1 = \{z : |z - 1| \leq |6| + |-1| = 7\}$$

$$C_2 = \{z : |z - (-2)| \leq |3| + |1| = 4\}$$

$$C_3 = \{z : |z - (-5)| \leq |-3| + |6| = 9\}$$

Gershgorincirklar och egenvärden

